**Setting up the Angular CLI workspace**

Setting up a frontend project today is more cumbersome than ever. We used to manually include the necessary JavaScript and CSS files in our HTML. Life used to be simple. Then frontend development became more ambitious: we started splitting our code into modules and using special tools called preprocessors for our code and CSS. Our projects became more complicated, and we started to rely on build systems to bundle our applications. As developers, we are not huge fans of configuration—we want to focus on building awesome apps. However, modern browsers do more to support the latest web standards, and some have even started to support JavaScript modules. That said, this is far from being widely supported. In the meantime, we still have to rely on bundling and module support tools. Setting up a project with Angular can be tricky. You need to know what libraries to import and ensure that files are processed in the correct order, which leads us to the topic of scaffolding. Scaffolder tools almost become necessary as complexity grows and where every hour counts towards producing business value rather than being spent fighting configuration problems. The primary motivation behind creating the Angular CLI was to help developers focus on application building, eliminating the configuration boilerplate. Essentially, with a simple command, you should be able to initialize an application, add new artifacts, run tests, and create a production-grade bundle. The Angular CLI supports all of this with the use of special commands.

**Prerequisites**

Before we begin, we need to ensure that our development environment includes a set of software tools essential to the Angular development workflow.

1. **Node.js**

Node.js is a JavaScript runtime built on top of Chrome’s v8 JavaScript engine. Angular requires an active or maintenance Long Time Support (LTS) version. If you have already installed it, you can run node -v in the command line to check which version you are running. The Angular CLI uses Node.js to accomplish specific tasks, such as serving, building, and bundling your application

1. **npm**

npm is a software package manager that is included by default in Node.js. You can check this out by running npm -v in the command line. An Angular application consists of various libraries, called packages, that exist in a central place called the npm registry. The npm client downloads and installs the libraries that are needed to run your application from the npm registry to your local computer

1. **Git**

Git is a client that allows us to connect to distributed version-control systems, such as GitHub, Bitbucket, and GitLab. It is optional from the perspective of the Angular CLI. You should install it if you want to upload your Angular project to a Git repository, which you might want to do.

1. **Angular CLI**

The Angular CLI is part of the Angular ecosystem and can be downloaded from the npm package registry. Since it is used for creating Angular projects, we need to install it globally in our system. Open a terminal and run the following command: npm install -g @angular/cli

**Angular CLI Commands**

• new (n): Creates a new Angular CLI workspace from scratch.

• build (b): Compiles an Angular application and outputs generated files in a predefined folder.

• generate (g): Creates new files that comprise an Angular application.

• serve (s): Builds an Angular application and serves it using a pre-configured web server.

• test (t): Runs unit tests of an Angular application.

• deploy: Deploys an Angular application to a web-hosting provider. You can choose from a collection of providers included in the Angular CLI.

• add: Installs an Angular library to an Angular application.

• completion: Enables auto-complete for Angular CLI commands through the terminal.

• update: Updates an Angular application to the latest Angular version

**Angular folder structure**

The workspace contains various folders and configuration files that the Angular CLI needs to build, test, and publish the Angular application:

• .vscode: Includes VS Code configuration files

• node\_modules: Includes npm packages needed for development and running the Angular application

• src: Contains the source files of the application

• .editorconfig: Defines coding styles for your editor

• .gitignore: Specifies files and folders that Git should not track

• angular.json: The main configuration file of the Angular CLI workspace

• package.json and package-lock.json: Provide definitions of npm packages, along with their exact versions, which are needed to develop, test, and run the Angular application

• README.md: A README file that is automatically generated from the Angular CLI

• tsconfig.app.json: TypeScript configuration that is specific to the Angular application

• tsconfig.json: TypeScript configuration that is specific to the Angular CLI workspace

• tsconfig.spec.json: TypeScript configuration that is specific to unit tests